9. A method of preparing software for a performance estimation, comprising:

Providing a software assembly code module;

Translating the assembly code module into a simulation model;

Annotating the simulation model with performance information;

10. The method of claim 9, wherein providing a software assembly code module comprises compiling software source code to assembly

11. The method of claim 10, wherein the software assembly code module is compiled using a compiler adapted to create code that will execute on a first machine architecture.

12. The method of claim 11, wherein the performance information is associated with the first machine architecture.

13. The method of claim 11, wherein the simulation model is compiled to execute on a second machine architecture, the second machine architecture being different from the first machine architecture.

14. The method of claim 9, wherein providing a software assembly code module comprises disassembling software binary code to assembly code.

15. The method of claim 9, wherein the simulation model is an assembler-level representation of the software, expressed in a high-level programming language.

16. The method of claim 9, wherein the translation step further comprises gathering information from another software module.

17. The method of claim 16, wherein the information gathered comprises high-level hints about the software assembly code module.

18. The method of claim 9, wherein the performance information comprises estimated performance information.

19. The method of claim 9, wherein the performance information is statically estimated.

20. The method of claim 9, wherein the performance information is dynamically computed at run-time, using a formula provided during the annotating step.

21. The method of claim 9, further comprising:

Compiling the simulation model to a simulator host program;

Executing the simulator host program on a simulator to allow performance measurements to be taken.

22. The method of claim 21, further comprising linking an already-annotated module with the simulation model.

23. A method of translating an assembly language software module into a simulation model, comprising;

Receiving the assembly language software module,

Parsing the assembly language software module into a data structure, the data structure comprising one or more nodes, each of the one or more nodes being mapped to a period of time using a mapping definition, each of the one or more nodes containing an element of the assembly language software module;

Processing the data structure to refine the accuracy of the simulation model;

Associating performance information with an element of the assembly language software module; and

Outputting the simulation model.

24. The method of claim 23, wherein the one or more nodes comprises a first node and a second node, the first node being mapped to a first period of time, the second node being mapped to a second period of time, the first period of time being different from the second period of time

25. The method of claim 23, wherein the performance information comprises an execution delay value for the element of the assembly language software module.

26. The method of claim 23, wherein the performance information is a statically computed value.

27. The method of claim 23, wherein the performance information is a formula for dynamically computing a value.

28. The method of claim 23, wherein processing the data structure comprises replicating the behavior of the assembly language software model in the simulation model.

29. A system for translating assembler code into a simulation model, comprising:

a data structure for storing the assembler code, the data structure comprising one or more nodes, each of the one or more nodes representing a unit of time, each of the one or more nodes for storing one or more elements of the assember code;

one or more operation modules for performing one or more operations on the assembler code stored in the data structure;

a machine description for providing information about a target machine architecture; and

a storage module for storing information related to the assembler code, the information being used by the one or more operations modules to perform one or more operations on the assembler code.

30. The system of claim 29, wherein the storage module is a symbol table.

31. The system of claim 29, wherein the one or more operations modules comprise a macro expansion module, a dependency resolution module, a symbol extraction module, a label identification module, a simulation model generation module, a simulation model assembly module, and a simulation model export module.

32. The system of claim 29, wherein the machine description is provided to an operation module.

33. A computer program product that includes a medium useable by a processor, the medium comprising a sequence of instructions which, when executed by said processor, causes said processor to execute a method for performing software performance analysis for a target machine, comprising:

describing a system design as a network of logical entities;

selecting at least one of the logical entities for a software implementation;

synthesizing a software program from the logical entities selected for the software implementation;

compiling the software program to generate an optimized assembler code representation of the software program;

performing a performance analysis using the assembler code;

generating a software simulation model using the assembler code; and

generating a hardware/software co-simulation model using the software simulation model.

34. The computer program product of claim 33, wherein the compiling step further comprises incorporating a description of the target machine.

35. The computer program product of claim 33, wherein the software simulation model is an assembler-level C code simulation model.

36. The computer program product of claim 33, further comprising selecting at least one of the logical entities for a hardware implementation, and synthesizing a software model of the hardware implementation from the selected logical entities, wherein the hardware/software co-simulation model is generated using the software model of the hardware implementation.

37. The computer program product of claim 33, wherein the performance analysis measures an execution time of an element of the assembler code.
38. The computer program product of claim 33, wherein the software program is compiled using the same compiler used to compile a production executable.
39. The computer program product of claim 33, wherein performing the performance analysis comprises annotating the assembler code with performance information.
40. The computer program product of claim 39, wherein the performance information is timing information.
41. A computer program product that includes a medium useable by a processor, the medium comprising a sequence of instructions which, when executed by said processor, causes said processor to execute a method for preparing software for a performance estimation, comprising:

    Providing a software assembly code module;

    Translating the assembly code module into a simulation model;

    Annotating the simulation model with performance information;
42. The computer program product of claim 41, wherein providing a software assembly code module comprises compiling software source code to assembly
43. The computer program product of claim 42, wherein the software assembly code module is compiled using a compiler adapted to create code that will execute on a first machine architecture.
44. The computer program product of claim 43, wherein the performance information is associated with the first machine architecture.
45. The computer program product of claim 43, wherein the simulation model is compiled to execute on a second machine architecture, the second machine architecture being different from the first machine architecture.
46. The computer program product of claim 41, wherein providing a software assembly code module comprises disassembling software binary code to assembly code.
47. The computer program product of claim 41, wherein the simulation model is an assembler-level representation of the software, expressed in a high-level programming language.
48. The computer program product of claim 41, wherein the translation step further comprises gathering information from another software module.

49. The computer program product of claim 48, wherein the information gathered comprises high-level hints about the software assembly code module.
50. The computer program product of claim 41, wherein the performance information comprises estimated performance information.
51. The computer program product of claim 41, wherein the performance information is statically estimated.
52. The computer program product of claim 41, wherein the performance information is dynamically computed at run-time, using a formula provided during the annotating step.
53. The computer program product of claim 41, further comprising:

Compiling the simulation model to a simulator host program;

Executing the simulator host program on a simulator to allow performance measurements to be taken.

54. The computer program product of claim 53, further comprising linking an already-annotated module with the simulation model.
55. A method of translating an assembly language software module into a simulation model, comprising;

Receiving the assembly language software module,

Parsing the assembly language software module into a data structure, the data structure comprising one or more nodes, each of the one or more nodes being mapped to a period of time using a mapping definition, each of the one or more nodes containing an element of the assembly language software module;

Processing the data structure to refine the accuracy of the simulation model;

Associating performance information with an element of the assembly language software module; and

Outputting the simulation model.

56. The computer program product of claim 55, wherein the one or more nodes comprises a first node and a second node, the first node being mapped to a first period of time, the second node being mapped to a second period of time, the first period of time being different from the second period of time
57. The computer program product of claim 55, wherein the performance information comprises an execution delay value for the element of the assembly language software module.

58. The computer program product of claim 55, wherein the performance information is a statically computed value.

59. The computer program product of claim 55, wherein the performance information is a formula for dynamically computing a value.

60. The computer program product of claim 55, wherein processing the data structure comprises replicating the behavior of the assembly language software model in the simulation model.